Test Protocol					
Test Case ID	**Test Case Description**	**Test Case Steps**	**Expected Result**	**Actual Result**	**Pass/Fail**
MCAL Module					
DIO Driver					
TC_DIO_001	Test dio pins to init output and input	Call DIO_init_pin()	all pins is working good	*Matches Expected Result*	Pass
TC_DIO_002	Test dio pins to write high and low	Call DIO_write_pin()	all pins is working good	*Matches Expected Result*	Pass
TC_DIO_003	Test dio pins init to read	Call DIO_read()	all pins is working good	*Matches Expected Result*	Pass
TC_DIO_004	Test dio pins init to toggle	Call DIO_toggle()	all pins is working good	*Matches Expected Result*	Pass
EXI Driver					
TC_EXI_001	Set External Interrupt Sence Control	Call EXT_vINTERRUPT_setSenseControl	Sence Control Set Sucessful For All Exti_interrupt	*Matches Expected Result*	Pass
TC_EXI_002	Set External Interrupt 0 Call Back	Call EXT0_INTERRUPT_SetInterruptHandler	Call Back Function Address Set Succesful For Exti_interrupt 0	*Matches Expected Result*	Pass
TC_EXI_003	Set External Interrupt 1 Call Back	Call EXT1_INTERRUPT_SetInterruptHandler	Call Back Function Address Set Succesful For Exti_interrupt 1	*Matches Expected Result*	Pass
TC_EXI_004	Set External Interrupt 2 Call Back	Call EXT2_INTERRUPT_SetInterruptHandler	Call Back Function Address Set Succesful For Exti_interrupt 2	*Matches Expected Result*	Pass
TC_EXI_005	Set All External Interrupt Call Back	Call EXT_INTERRUPT_SetInterruptHandler	Call Back Function Address Set Succesful For All Exti_interrupt	*Matches Expected Result*	Pass
TC_EXI_006	Stop All Interrupts	Call EXT_vINTERRUPT_Denit	Stop All Exti_interrupt	*Matches Expected Result*	Pass
TC_EXI_007	Set Sence Control & Call Back	Call EXT_vINTERRUPT_Init	Set Sence Control & Call Back Function For All Exti_interrupt	*Matches Expected Result*	Pass
TIMER Driver					
TC_TIMER_001	Test timer0 & timer2 Init	1. Call TMR_timer0NormalModeInit & TMR_timer2NormalModeInit	The two timers working in the normal mode	*Matches Expected Result*	Pass
TC_TIMER_002	Test timer0 delay function	1. Call TMR_timer2Delay with deferent delay ranges	The desired delay is acheived	*Matches Expected Result*	Pass
TC_TIMER_003	Test timer2 delay function	1. Call TMR_timer0Delay with deferent delay ranges	The desired delay is acheived	*Matches Expected Result*	Pass
TC_TIMER_004	Test timer0 & timer2 start	1. Call TMR_timer0Start & TMR_timer2Start	The two timers start working in normal mode	*Matches Expected Result*	Pass
PWM Driver					
TC_PWM_001	Timer0 initialization	Call Timer0_init	the timer initializes in Normal Mode	*Matches Expected Result*	Pass
TC_PWM_002	Timer0 start	Call TIMER0_start	the timer starts counting	*Matches Expected Result*	Pass
TC_PWM_003	Timer0 Stop	Call TIMER0_stop	the timer stops counting	*Matches Expected Result*	Pass
TC_PWM_004	TIMER0 initializes PWM	Call TIMER0_initPWM	initializes all pwm pins correctly	*Matches Expected Result*	Pass
TC_PWM_005	TIMER0 set Pwm	Call TIMER0_setPwm & send dutyCycle as argument	the wave generated with desired dutyCycle	*Matches Expected Result*	Pass
TC_PWM_006	TIMER0 PWM ExecutedFunction	This Function Called inside ISR	it flips the level of cycle	*Matches Expected Result*	Pass
ICU Driver					
TC_ICU_001	Test ICU_captureRisingEdge	1. Call ICU_captureRisingEdge	The ICU can capture Rising Edge	*Matches Expected Result*	Pass
TC_ICU_002	Test ICU_timer1NormalModeInit	1. Call TMR_timer1NormalModeInit	The timer1 working in the normal mode	*Matches Expected Result*	Pass
TC_ICU_003	Test ICU_timer1_ISR	1. Call Enable Timer1_ISR	The timer1_ISR working	*Matches Expected Result*	Pass
TC_ICU_004	Test ICU_exti2Enable	1. Call Enable EXTI_2 To Fire ISR On Falling & Rising Sens	The EXTI_2 is working as required	*Matches Expected Result*	Pass
HAL Module					
DCM Driver					
TC_DCM_001	Test DCM_motorInit	1. Call DCM_motorInit	The motor pins are initialized to be output	*Matches Expected Result*	Pass
TC_DCM_002	Test DCM_changeDirescction	1. Call DCM_changeDirection	The motor direction is changed	*Matches Expected Result*	Pass
TC_DCM_003	Test DCM_vdStop	1. Call DCM_vdStop	The motors are stopped	*Matches Expected Result*	Pass
TC_DCM_004	Test DCM_setDutyCycle	1. Call DCM_setDutyCycle	The duty Cycle is passed to the PWM function	*Matches Expected Result*	Pass
TC_DCM_005	Test DCM_rotate	1. Call DCM_setDutyCycle	The robot will rotate	*Matches Expected Result*	Pass
P_Bttn Driver					
TC_PBTN_001	Intialize Push Button Pins	Call PUSH_BTN_intialize To Intialize Push Button Pins	All Configration Intialize Succesful	*Matches Expected Result*	Pass
TC_PBTN_002	Get Push Button Status	Call PUSH_BTN_read_state To Get Its Status Pressed Or Relased	Push Button Status Returned Succesful	*Matches Expected Result*	Pass
LCD Driver					
TC_LCD_001	Test LCD commands	call LCD_write_ins	LCD command working good	*Matches Expected Result*	Pass
TC_LCD_002	Test LCD write string	call LCD_write_string	LCD write working good	*Matches Expected Result*	Pass
TC_LCD_003	Test LCD write number	Call LCD_write _number	LCD write working good	*Matches Expected Result*	Pass
TC_LCD_004	Test LCD Set cursor	Call LCD_Set_cursor	LCD set cursor working good	*Matches Expected Result*	Pass
KPD Driver					
TC_KPD_001	Keypad Initialization	Call KEYPAD_init	the pins initializes correctly	*Matches Expected Result*	Pass
TC_KPD_002	Keypad GET BUTTON	Call KEYPAD_GetButton and press Nothing	Nothing Press returned from function	*Matches Expected Result*	Pass
TC_KPD_003	Keypad GET BUTTON	Call KEYPAD_GetButton and press first key	key 1 returned from function	*Matches Expected Result*	Pass
TC_KPD_004	Keypad GET BUTTON	Call KEYPAD_GetButton and press second key	Key 2 returned from function	*Matches Expected Result*	Pass
Ultrasonic Driver					
TC_ULTRASONIC_001	Intialize Pins & Enable Global Interrupts	Call ultrasonic_vInit	Ultrasonic Pins & Configuration Intialize Succesful	*Matches Expected Result*	Pass
TC_ULTRASONIC_002	Start Trigger & Get Distance In Cm	Call ultrasonic_vGetDistance	Ultrasonice Trigger & Get The Distance In Cm	*Matches Expected Result*	Pass
Call ultrasonic_vGetDistance					

APP					
TC_APP_001	initialize all Hal Modules	Call app_init	all modules initializes correctly	*Matches Expected Result*	Pass
TC_APP_002	Run main Logic of application	Call app_main in super loop	app works fine and covered all known cases	*Matches Expected Result*	Pass
User Stories					
TC_ObstcaleAvoidance_APP_001	First start pressed	press key 1 on keybad	the app starts correctly and asks user to setDefault rotation	*Matches Expected Result*	Pass
TC_ObstcaleAvoidance_APP_002	set Default Rotation	press push Button to change the rotation settings for 5 seconds	the rotation changes every time pb pressed during 5 seconds	*Matches Expected Result*	Pass
TC_ObstcaleAvoidance_APP_003	Obstacle more than 70 case	robot read distance far than 70	it starts with 30 % speed for 5 seconds then increase it to 50 %	*Matches Expected Result*	Pass
TC_ObstcaleAvoidance_APP_004	Obstacle more than 30 case	robot read distance far than 30 and less than 70	the robot moves with 30%	*Matches Expected Result*	Pass
TC_ObstcaleAvoidance_APP_005	Obstacle more than 20 case	robot read distance far than 20 and less tnan 30	the robot stops then rotate and check the all directions	*Matches Expected Result*	Pass
TC_ObstcaleAvoidance_APP_006	Obstacle less than 20	robot read distance less than 20	the robot stops and move backward with 30 % speed	*Matches Expected Result*	Pass
TC_ObstcaleAvoidance_APP_007	Stop pressed	press key 2 on keypad	the robot sttoped	*Matches Expected Result*	Pass
TC_ObstcaleAvoidance_APP_008	start pressed after stop	the robot read distance and jump to any case depend on distance	the robot starts with scanning distance	*Matches Expected Result*	Pass
TC_ObstcaleAvoidance_APP_009	Press Stop Without Starting The Robot	User Press Stop Button When The Robot Is Stopped	Nothing Will Happen	*Matches Expected Result*	Pass
TC_ObstcaleAvoidance_APP_010	Press Start When The Robot Ruuning	User Press Starting Button Again When The Robot Is Running	Nothing Will Happen	*Matches Expected Result*	Pass
TC_ObstcaleAvoidance_APP_011	Obstacle Is Surrounding The Rebot	Rebot Will Rotate In All Direction Every 3 Seconds To Check Any Possible Direction	Robot Do Steps Secusseful	*Matches Expected Result*	Pass